UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
09/771,761	01/29/2001	Jeff A. Zimniewicz	MS160268.1	8645

27195 7590 12/09/2003

AMIN & TUROCY, LLP
24TH FLOOR, NATIONAL CITY CENTER
1900 EAST NINTH STREET
CLEVELAND, OH 44114

EXAMINER	
YIGDALL, MICHAEL J	
ART UNIT	**PAPER NUMBER**
2122	3

DATE MAILED: 12/09/2003

Please find below and/or attached an Office communication concerning this application or proceeding.

Office Action Summary	Application No.	Applicant(s)	
	09/771,761	ZIMNIEWICZ ET AL.	
	Examiner	Art Unit	
	Michael J. Yigdall	2122	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If the period for reply specified above is less than thirty (30) days, a reply within the statutory minimum of thirty (30) days will be considered timely.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133).
- Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on *29 January 2001*.

2a)☐ This action is **FINAL**. 2b)☒ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) *1-31* is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) *1-31* is/are rejected.

7)☐ Claim(s) _____ is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☒ The specification is objected to by the Examiner.

10)☒ The drawing(s) filed on *29 January 2001* is/are: a)☒ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. §§ 119 and 120

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

13)☐ Acknowledgment is made of a claim for domestic priority under 35 U.S.C. § 119(e) (to a provisional application) since a specific reference was included in the first sentence of the specification or in an Application Data Sheet. 37 CFR 1.78.

a) ☐ The translation of the foreign language provisional application has been received.

14)☐ Acknowledgment is made of a claim for domestic priority under 35 U.S.C. §§ 120 and/or 121 since a specific reference was included in the first sentence of the specification or in an Application Data Sheet. 37 CFR 1.78.

Attachment(s)

1) ☒ Notice of References Cited (PTO-892)
2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)
3) ☒ Information Disclosure Statement(s) (PTO-1449) Paper No(s) *2*.
4) ☐ Interview Summary (PTO-413) Paper No(s). _____.
5) ☐ Notice of Informal Patent Application (PTO-152)
6) ☐ Other: .

DETAILED ACTION

1. Claims 1-31 are pending and have been examined. The priority date considered for the application is 29 January 2001.

Specification

2. The disclosure is objected to because of the following informalities: The specification contains typographical errors. See, for example, "associated each" in paragraph 12, which should read --associated with each--; also see, for example, "appreciated" in paragraph 65, which should read --appreciate--. Appropriate correction is required.

Claim Objections

3. Claim 8 is objected to because of the following informalities: The phrase "component component" in line 4 should be replaced with --component--. Appropriate correction is required. The claim has been interpreted assuming this correction to be made.

Claim Rejections - 35 USC § 112

4. The following is a quotation of the second paragraph of 35 U.S.C. 112:

> The specification shall conclude with one or more claims particularly pointing out and distinctly claiming the subject matter which the applicant regards as his invention.

5. Claims 22 and 24 are rejected under 35 U.S.C. 112, second paragraph, as being indefinite for failing to particularly point out and distinctly claim the subject matter which applicant regards as the invention.

Claim 22 recites the limitation "the setup manager" in line 1. There is insufficient antecedent basis for this limitation in the claim. The claim was perhaps intended to be dependent upon claim 13 rather than upon claim 1.

Claim 24 recites the limitation "the setup manager" in line 10. There is insufficient antecedent basis for this limitation in the claim. The claim recites both a dependency manager and a setup engine, but not a setup manager.

Claim Rejections - 35 USC § 102

6. The following is a quotation of the appropriate paragraphs of 35 U.S.C. 102 that form the basis for the rejections under this section made in this Office action:

> A person shall be entitled to a patent unless –
>
> (e) the invention was described in (1) an application for patent, published under section 122(b), by another filed in the United States before the invention by the applicant for patent or (2) a patent granted on an application for patent by another filed in the United States before the invention by the applicant for patent, except that an international application filed under the treaty defined in section 351(a) shall have the effects for purposes of this subsection of an application filed in the United States only if the international application designated the United States and was published under Article 21(2) of such treaty in the English language.

7. Claims 1, 8-13, 16, 18, 21, 22, 25 and 26 are rejected under 35 U.S.C. 102(e) as being anticipated by U.S. Pat. No. 6,442,754 to Curtis.

With respect to claim 1, Curtis discloses a system to facilitate installation and/or removal of components (see the title and abstract) including at least one shared component (see column 4, lines 33-37, which shows components that are depended upon by more than one program, i.e. shared components), comprising:

(a) a validation engine operative to provide a valid order (see column 11, lines 11-20, which shows a function for checking dependencies, i.e. a validation engine; see also column 12, lines 22-32, which shows generating a list of dependent components and providing a valid installation order); and

(b) an installer operative to control at least one of an install and removal operation of the components based on the valid order and operative to effect manipulation of at least one property

associated with the at least one shared component to reflect dependency for the at least one shared component according to the installation or removal thereof (see column 12, lines 32-50, which shows an installer for installing the components based on the valid order; see also Fig. 5 and column 13, lines 7-10, which shows a data structure having properties that reflect dependency; see also column 13, lines 28-29, which shows manipulating the data structure when a component is installed).

With respect to claim 8, Curtis further discloses the limitation wherein the at least one property further comprises configuration data indicative of an operating relationship of the at least one shared component and each installed dependent component associated with the at least one shared component (see Fig. 5 and column 13, lines 7-27, which shows a data structure having properties indicative of the relationship between a component and its dependencies).

With respect to claim 9, Curtis further discloses the limitation wherein the installer is operative to control installation of the at least one shared component, such that a single set of files for the at least one shared component is copied as part of the installation for use by associated dependent components (see column 9, lines 47-64, which shows determining whether dependencies are already installed and installing a set of files for a shared component).

With respect to claim 10, Curtis further discloses the limitation wherein the at least one shared component has associated metadata operable to identify the at least one shared component as a shared component (see Fig. 3 and column 9, lines 10-25, which shows a dependency object comprising metadata that identifies whether a component is a shared component).

With respect to claim 11, Curtis further discloses the limitation wherein the at least one shared component requires at least one dependent component to perform a substantially useful

function (see column 9, lines 25-31, which shows that dependent components must be installed in order for another component to perform all intended functions).

With respect to claim 12, Curtis further discloses the limitation wherein a runtime dependency exists between an installed dependent component and the shared component on which the dependent component depends (see column 9, lines 39-43, which shows dependencies needed by a component in order to operate, i.e. dependencies needed at runtime).

With respect to claim 13, Curtis further discloses a system to facilitate installation of components (see the title and abstract) including at least one shared component (see column 4, lines 33-37, which shows components that are depended upon by more than one program, i.e. shared components), comprising:

(a) a setup manager which controls installation of the components (see column 5, lines 56-60, which shows an installer script, i.e. a setup manager);

(b) dependency manager which provides a valid installation order based on metadata associated with at least some of the components (see column 11, lines 11-20, which shows a function for checking dependencies, i.e. a dependency manager, using dependency objects; see also Fig. 3 and column 9, lines 10-25, which shows that the dependency objects comprise metadata; see also column 12, lines 22-32, which shows generating a list of dependent components and providing a valid installation order);

wherein the setup manager causes the components to be installed according to the valid installation order, a separate shared installation of the at least one shared component being implemented for each dependent component that depends on the at least one shared component

(see column 12, lines 32-50, which shows installing each component based on the valid installation order).

With respect to claim 16, see the explanation for claim 10 above.

With respect to claim 18, Curtis further discloses at least one property associated with an installed instance of the at least one shared component which reflects dependency for the at least one shared component (see Fig. 5 and column 13, lines 7-10, which shows a data structure having properties that reflect dependency).

With respect to claim 21, see the explanation for claim 8 above.

With respect to claim 22, see the explanation for claim 9 above.

With respect to claim 25, Curtis discloses a system to facilitate installation and/or removal of components (see the title and abstract) including at least one shared component (see column 4, lines 33-37, which shows components that are depended upon by more than one program, i.e. shared components), comprising:

(a) means for providing a valid order for the components (see column 12, lines 22-32, which shows generating a list of dependent components and providing a valid installation order);

(b) means for controlling installation of the components based on the valid order (see column 12, lines 32-50, which shows an installer for installing the components based on the valid order); and

(c) means for manipulating at least one property associated with the at least one shared component to reflect dependency for the at least one shared component based on at least one installation of the shared component and removal of a dependent component that depends on the at least one shared component (see Fig. 5 and column 13, lines 7-10, which shows a data

structure having properties that reflect dependency; see also column 13, lines 28-29, which shows manipulating the data structure when a component is installed).

With respect to claim 26, Curtis discloses a method to facilitate installing and/or removing components (see the title and abstract) including at least one shared component (see column 4, lines 33-37, which shows components that are depended upon by more than one program, i.e. shared components), the method comprising:

(a) providing a valid order (see column 12, lines 22-32, which shows generating a list of dependent components and providing a valid installation order);

(b) installing each of the plurality of components based on the valid order (see column 12, lines 32-50, which shows installing the components based on the valid order); and

(c) modifying at least one property associated with the at least one shared component to reflect dependency characteristics of the at least one shared component relative dependent components operative to use the at least one shared component (see Fig. 5 and column 13, lines 7-10, which shows a data structure having properties that reflect dependency; see also column 13, lines 28-29, which shows manipulating the data structure when a component is installed).

Claim Rejections - 35 USC § 103

8. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

9. Claim 2-4, 14, 15, 17, 23, 24 and 29-31 are rejected under 35 U.S.C. 103(a) as being unpatentable over Curtis as applied to claims 1, 8-13, 16, 18, 21, 22, 25 and 26 above, in view of U.S. Pat. No. 5,721,824 to Taylor.

With respect to claim 2, Curtis does not disclose the limitation wherein the valid order identifies shared components for installation subsequent to non-shared components.

Curtis shows that shared, or dependent, components are identified for installation prior to non-shared components (see column 12, lines 27-32).

Taylor discloses the limitation above in terms of an action list, i.e. a valid order, that identifies shared, or dependent, components for installation subsequent to a non-shared package (see column 5, lines 25-29). Note that Taylor also discloses an implementation wherein dependent components are identified for installation before non-shared components, as in the Curtis system (see column 7, lines 49-53).

It would have been obvious to one of ordinary skill in the art at the time the invention was made to modify the Curtis system with the feature of identifying shared components for installation subsequent to non-shared components, as taught by Taylor, for the purpose of supporting an installation sequence that conforms to the constraints of the target system (see Taylor, column 2, lines 1-3), in order to increase the compatibility of the installation routine with different platforms.

With respect to claim 3, Curtis further discloses the limitation wherein the installer is operative to initiate a method to install each of the components based on the valid order during a part of the installation (see column 12, lines 32-50, which shows an installer for installing each

of the components based on the valid order; see also column 12, lines 59-62, which shows that the installer is operative to initiate the installation).

Curtis does not disclose the limitation wherein the at least one shared component is installed and configured for a selected dependent component during the first part of installation.

Taylor further discloses the limitation above in terms of installing components based on the action list, i.e. the valid order, during a first part of the installation (see column 2, lines 7-11, which shows that the flow of operations is layered, i.e. has multiple parts; see also column 2, lines 12-26, which shows installing packages during a first part of the installation).

It would have been obvious to one of ordinary skill in the art at the time the invention was made to modify the Curtis system with the feature of installing components during a first part of the installation, as taught by Taylor, for the purpose of installing multiple software packages with a single load on the system, in order to improve the perceived performance (see Taylor, column 3, lines 47-51).

With respect to claim 4, Curtis does not disclose the limitation wherein the method is operative to employ a second part of the installation to install the at least one shared component for each dependent component other than the selected dependent component during the second part of the installation.

Taylor further discloses the limitation above in terms of installing packages or components that are depended upon by other dependent packages during a second part of the installation (see column 2, lines 7-11, which shows that the flow of operations is layered, i.e. has multiple parts; see also column 2, lines 53-62, which shows installing packages during a second part of the installation).

It would have been obvious to one of ordinary skill in the art at the time the invention was made to modify the Curtis system with the feature of installing components during a second part of the installation, as taught by Taylor, for the purpose of installing multiple software packages with a single load on the system, in order to improve the perceived performance (see Taylor, column 3, lines 47-51).

With respect to claim 14, Curtis does not disclose the limitation wherein the dependency manager is operative to validate a received installation order, which, upon validation of the received installation order, becomes the valid installation order.

Curtis does show a dependency manager for generating a valid installation order (see column 11, lines 11-20, which shows a function for checking dependencies, i.e. a dependency manager; see also column 12, lines 22-32, which shows generating a list of dependent components and providing a valid installation order).

Taylor further discloses the limitation above in terms of validating a dependency list, i.e. a received installation order, and using it as the valid installation order (see column 2, lines 28-40, which shows translating the dependency list into an action list, i.e. a valid installation order).

It would have been obvious to one of ordinary skill in the art at the time the invention was made to modify the Curtis system with the feature of validating an installation order, as taught by Taylor, for the purpose of installing multiple software packages with a single load on the system, in order to improve the perceived performance (see Taylor, column 3, lines 47-51).

With respect to claim 15, Curtis does not disclose the limitation wherein, if the received installation order is improper, the dependency manager is operative to create the valid installation order.

Curtis does show a dependency manager for generating a valid installation order (see column 11, lines 11-20, which shows a function for checking dependencies, i.e. a dependency manager; see also column 12, lines 22-32, which shows generating a list of dependent components and providing a valid installation order).

Taylor further discloses the limitation above in terms of validating a dependency list, i.e. a received installation order, and creating a valid installation order (see column 2, lines 12-26, which shows generating an action list, i.e. a valid installation order).

It would have been obvious to one of ordinary skill in the art at the time the invention was made to modify the Curtis system with the feature of validating an installation order, as taught by Taylor, for the purpose of installing multiple software packages with a single load on the system, in order to improve the perceived performance (see Taylor, column 3, lines 47-51).

With respect to claim 17, Curtis does not disclose the limitation wherein the setup manager is operative to initiate a method to install each of the components according to the valid installation order during a first part of the installation, the at least one shared component being installed for a first dependent component during the first part of installation, the method being operative to install the at least one shared component for each other dependent component during a second part of the installation.

Curtis does show an installer or setup manager for installing each of the components based on the valid installation order (see column 12, lines 32-50; see also column 12, lines 59-62, which shows that the installer is operative to initiate the installation).

Taylor further discloses the limitation above in terms of installing components based on the action list, i.e. the valid installation order, during a first part of the installation, and installing

components that are depended upon by other dependent packages during a second part of the installation (see column 2, lines 7-11, which shows that the flow of operations is layered, i.e. has multiple parts; see also column 2, lines 12-26, which shows installing packages during a first part of the installation; see also column 2, lines 53-62, which shows installing packages during a second part of the installation).

It would have been obvious to one of ordinary skill in the art at the time the invention was made to modify the Curtis system with the feature of first and second installation parts, as taught by Taylor, for the purpose of installing multiple software packages with a single load on the system, in order to improve the perceived performance (see Taylor, column 3, lines 47-51).

With respect to claim 23, Curtis discloses a system to facilitate installation and/or removal of components (see the title and abstract) including at least one shared component (see column 4, lines 33-37, which shows components that are depended upon by more than one program, i.e. shared components), comprising:

(a) a validation component operative to provide a valid order based on setup data (see column 11, lines 11-20, which shows a function for checking dependencies, i.e. a validation component, using dependency objects; see also Fig. 3 and column 9, lines 10-25, which shows that the dependency objects comprise setup data; see also column 12, lines 22-32, which shows generating a list of dependent components and providing a valid installation order).

Curtis does not disclose the limitation of:

(b) a setup engine operative to initiate installation of each of the components according to the valid order during a first part of the installation, the shared component being installed for a first dependent component during the first part of installation, the shared component being

installed for each other dependent component during a second part of the installation separate from the first part.

Curtis does show an installer or setup engine for installing each of the components based on the valid installation order (see column 12, lines 32-50; see also column 12, lines 59-62, which shows that the installer is operative to initiate the installation).

Taylor discloses the limitation above in terms of installing components based on the action list, i.e. the valid installation order, during a first part of the installation, and installing components that are depended upon by other dependent packages during a second part of the installation (see column 2, lines 7-11, which shows that the flow of operations is layered, i.e. has multiple parts; see also column 2, lines 12-26, which shows installing packages during a first part of the installation; see also column 2, lines 53-62, which shows installing packages during a second part of the installation).

It would have been obvious to one of ordinary skill in the art at the time the invention was made to modify the Curtis system with the feature of first and second installation parts, as taught by Taylor, for the purpose of installing multiple software packages with a single load on the system, in order to improve the perceived performance (see Taylor, column 3, lines 47-51).

With respect to claim 24, Curtis discloses a system to facilitate installation and/or removal of components (see the title and abstract) including at least one shared component (see column 4, lines 33-37, which shows components that are depended upon by more than one program, i.e. shared components), comprising:

(a) a dependency manager operative to provide a valid order based on setup data (see column 11, lines 11-20, which shows a function for checking dependencies, i.e. a dependency

manager, using dependency objects; see also Fig. 3 and column 9, lines 10-25, which shows that the dependency objects comprise setup data; see also column 12, lines 22-32, which shows generating a list of dependent components and providing a valid installation order).

Curtis does not disclose the limitation of:

(b) a setup engine operative to initiate installation of each of the components according to the valid order during a first part of the installation, the shared component being installed for a first dependent component during the first part of installation, the shared component being installed for each other dependent component during a second part of the installation, which is subsequent to the first part.

Curtis does show an installer or setup engine for installing each of the components based on the valid installation order (see column 12, lines 32-50; see also column 12, lines 59-62, which shows that the installer is operative to initiate the installation).

Taylor discloses the limitation above in terms of installing components based on the action list, i.e. the valid installation order, during a first part of the installation, and installing components that are depended upon by other dependent packages during a second part of the installation (see column 2, lines 7-11, which shows that the flow of operations is layered, i.e. has multiple parts; see also column 2, lines 12-26, which shows installing packages during a first part of the installation; see also column 2, lines 53-62, which shows installing packages during a second part of the installation).

Curtis further discloses the limitation wherein the setup manager is operative to effect manipulation of at least one property associated with the at least one shared component to reflect dependency characteristics of the at least one shared component as a function of at least one of

installation of the shared component and removal of a dependent component that depends on the at least one shared component (see Fig. 5 and column 13, lines 7-10, which shows a data structure having properties that reflect dependency characteristics; see also column 13, lines 28-29, which shows manipulating the data structure when a component is installed).

It would have been obvious to one of ordinary skill in the art at the time the invention was made to modify the Curtis system with the feature of first and second installation parts, as taught by Taylor, for the purpose of installing multiple software packages with a single load on the system, in order to improve the perceived performance (see Taylor, column 3, lines 47-51).

With respect to claim 29, see the explanation for claim 3 above.

With respect to claim 30, see the explanation for claim 4 above.

With respect to claim 31, Curtis discloses a method to facilitate installing and/or removing components including at least one shared component, the method comprising:

(a) providing a valid order (see column 12, lines 22-32, which shows generating a list of dependent components and providing a valid installation order).

Curtis does not disclose the steps of:

(b) effecting installation of each of the components during a first part of installation according to the valid order, the shared component being installed for a first dependent component during the first part of the installation;

(c) effecting installation of the shared component for each other dependent component during a second part of the installation separate from the first part.

Curtis does show an installer for installing each of the components based on the valid order (see column 12, lines 32-50; see also column 12, lines 59-62, which shows that the installer is operative to initiate the installation).

Taylor discloses step (b) above in terms of installing components based on the action list, i.e. the valid order, during a first part of the installation (see column 2, lines 7-11, which shows that the flow of operations is layered, i.e. has multiple parts; see also column 2, lines 12-26, which shows installing packages during a first part of the installation).

Taylor further discloses step (c) above in terms of installing packages or components that are depended upon by other dependent packages during a second part of the installation (see column 2, lines 7-11, which shows that the flow of operations is layered, i.e. has multiple parts; see also column 2, lines 53-62, which shows installing packages during a second part of the installation).

It would have been obvious to one of ordinary skill in the art at the time the invention was made to modify the Curtis system with the feature of first and second installation parts, as taught by Taylor, for the purpose of installing multiple software packages with a single load on the system, in order to improve the perceived performance (see Taylor, column 3, lines 47-51).

10. Claims 5-7, 19, 20, 27 and 28 are rejected under 35 U.S.C. 103(a) as being unpatentable over Curtis as applied to claims 1, 8-13, 16, 18, 21, 22, 25 and 26 above, in view of U.S. Pat. No. 6,367,075 to Kruger et al.

With respect to claim 5, Curtis does not disclose the limitation wherein the at least one property further comprises a reference count having a value indicative of a number of dependent components associated with the at least one shared component.

Curtis does show storing dependency information in order to indicate which components depend on a shared component (see column 13, lines 1-6).

Kruger et al. discloses the limitation above in terms of an installer that uses a reference count for shared library files, to ensure that files depended upon by other programs are not affected (see column 9, lines 14-21).

It would have been obvious to one of ordinary skill in the art at the time the invention was made to modify the Curtis system with the reference count feature taught by Kruger et al., for the purpose of ensuring that shared components used by other programs are not affected, inherently reducing the number of potential version conflicts.

With respect to claim 6, Curtis does not disclose the limitation wherein the installer is operative to effect an increase in the value of the reference count for each installation of the at least one shared component.

Curtis does show storing dependency information in order to indicate which components depend on a shared component (see column 13, lines 1-6; note that the information is written during installation).

Kruger et al. further discloses the limitation above in terms of incrementing the reference count when a file is added or installed, to ensure that files depended upon by other programs are not affected (see column 9, lines 14-21).

It would have been obvious to one of ordinary skill in the art at the time the invention was made to modify the Curtis system with the reference count feature taught by Kruger et al., for the purpose of ensuring that shared components used by other programs are not affected, inherently reducing the number of potential version conflicts.

With respect to claim 7, Curtis does not disclose the limitation wherein the installer is operative to effect a decrease in the value of the reference count in response to removal of a dependent component that depends on the at least one shared component.

Curtis does show storing dependency information in order to indicate which components depend on a shared component (see column 13, lines 1-6; note that the information is used when a component is to be uninstalled).

Kruger et al. further discloses the limitation above in terms of decreasing the reference count when a file is deleted or removed, to ensure that files depended upon by other programs are not affected (see column 9, lines 14-21).

It would have been obvious to one of ordinary skill in the art at the time the invention was made to modify the Curtis system with the reference count feature taught by Kruger et al., for the purpose of ensuring that shared components used by other programs are not affected, inherently reducing the number of potential version conflicts.

With respect to claim 19, see the explanation for claim 5 above.

With respect to claim 20, see the explanations for claims 6 and 7 above.

With respect to claim 27, see the explanation for claim 5 above.

With respect to claim 28, see the explanations for claims 6 and 7 above.

Conclusion

11. The prior art made of record and not relied upon is considered pertinent to applicant's disclosure. U.S. Pat. No. 6,381,742 to Forbes et al. discloses a software package management system for installing and removing components.

Any inquiry concerning this communication or earlier communications from the examiner should be directed to Michael J. Yigdall whose telephone number is (703) 305-0352. The examiner can normally be reached on Monday through Friday from 8:00am to 4:30pm.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Tuan Q. Dam, can be reached on (703) 305-4552. The fax phone number for the organization where this application or proceeding is assigned is (703) 746-7239.

Any inquiry of a general nature or relating to the status of this application or proceeding should be directed to the receptionist whose telephone number is (703) 305-3900.

MY

Michael J. Yigdall
Examiner
Art Unit 2122

mjy
December 3, 2003

JOHN CHAVIS
PATENT EXAMINER
ART UNIT 2124